import java.awt.Rectangle;

import java.awt.Polygon;

/\*\*

\* LaserShoot Class calculates the new X and Y moving angle of laser by

\* converting the degrees to radians and then using sine to get the vertical

\* distance and cosine for the horizontal distance.

\*

\* @author Ram Reddy & Eric Feng

\* @version 5.24.20

\*/

public class LaserShoot extends Polygon {

// height of Game window

int heightOfWindow = AsteroidGame.windowHeight;

// width of Game window

int widthOfWindow = AsteroidGame.windowWidth;

// variable laserX initialized value to 0 for x coordinate

// variable laserY initialized value to 0 for y coordinate

private double laserX = 0, laserY = 0;

// Creates the array polyXValues to set X coordinates of laser

public static int[] polyXValues = { -3, 3, 3, -3, -3 };

// Creates the array polyYValues and initialized Y coordinate points for laser

public static int[] polyYValues = { -3, -3, 3, 3, -3 };

// Creates the variable laserW and laserH to set width and height of laser

// Sets width and height to 6 pixels

// Set dimensions for laser

private int laserW = 6, laserH = 6;

// boolean laserOnScreen to tell weather laser is on screen or not

// laserOnScreen initialized to false

public boolean laserOnScreen = false;

// initialized xVelocity and yVelocity, velocity of laser to 5

private double xVelocity = 5, yVelocity = 5;

// variable laserAngle, set initial angle to 0

private double laserAngle = 0;

/\*\*

\* Constructor sets the center, velocity, and visibility Since laser is present

\* on the screen, sets laserOnScreen flag to true

\*

\* @param laserX Assigning new value of laserX to laserX

\* @param laserY Assigning new value of laserY to laserY

\* @param laserAngle Assigning new value of laserAngle to laserAngle

\*/

public LaserShoot(double laserX, double laserY, double laserAngle) {

// sets laser trajectory based on moving angle and velocity of the laser

super(polyXValues, polyYValues, 5);

this.laserX = laserX;

this.laserY = laserY;

this.laserAngle = laserAngle;

this.laserOnScreen = true;

this.setXVelocity(this.laserXMovingAngle(laserAngle) \* 10);

this.setYVelocity(this.laserYMovingAngle(laserAngle) \* 10);

}

// Setter and Getter methods for Laser X and Y

/\*\*

\* Getting current laser x coordinate and returning the value

\*

\* @return laserX

\*/

public double getlaserX1() {

return this.laserX;

}

/\*\*

\* Getting current laser y coordinate and returning the value

\*

\* @return laserY

\*/

public double getlaserY1() {

return this.laserY;

}

/\*\*

\* Setting laserX coordinates

\*

\* @param laserX Changing laserX to new value

\*/

public void setlaserX1(double LaserX) {

this.laserX = LaserX;

}

/\*\*

\* Setting laserY coordinate to the new LaserY value

\*

\* @param laserY Changing laserY to new value

\*/

public void setlaserY1(double LaserY) {

this.laserY = LaserY;

}

/\*\*

\* Get the current x velocity value

\*

\* @return xVelocity - Laser X velocity

\*/

public double getXVelocity() {

return this.xVelocity;

}

/\*\*

\* Get the current y velocity value

\*

\* @return yVelocity - Laser Y velocity

\*/

public double getYVelocity() {

return this.yVelocity;

}

/\*\*

\* Set new XVelocity value

\*

\* @param xVelocity Changing xVelocity to newest value

\*/

public void setXVelocity(double xVelocity) {

this.xVelocity = xVelocity;

}

/\*\*

\* Set new YVelocity value

\*

\* @param yVelocity Changing yVelocity to newest value

\*/

public void setYVelocity(double yVelocity) {

this.yVelocity = yVelocity;

}

// Double the x moving angle with the formula with center as base.

// Double the x moving angle with the formula with center as base.

/\*\*

\* Get current laser height value

\*

\* @return laserH - Height of the laser

\*/

public int getH() {

return this.laserH;

}

/\*\*

\* Get the current laser width value

\*

\* @return laserW Width of the laser.

\*/

public int getW() {

return this.laserW;

}

/\*\*

\* Set new laserAngle value

\*

\* @param laserAngle Changing laserAngle to new value

\*/

public void setLaserAngle(double laserAngle) {

this.laserAngle = laserAngle;

}

/\*\*

\* Get current laserAngle value

\*

\* @return laserAngle

\*/

public double getLaserAngle() {

return this.laserAngle;

}

/\*\*

\* increase current laserX value by increaseValue parameter value

\*

\* @param increaseValue increase laserX by increaseValue

\*/

public void newXPosition(double increaseValue) {

this.laserX += increaseValue;

}

/\*\*

\* increase current laserY value by increaseValue parameter value

\*

\* @param laserY increase laserY by increaseValue

\*/

public void newYPosition(double increaseValue) {

this.laserY += increaseValue;

}

/\*\*

\* After all the getter and setter methods we are finally returning all laser

\* information

\*

\* @return Rectangle Rectangle object

\*/

public Rectangle getBounds() {

return new Rectangle((int) getlaserX1() - 6, (int) getlaserY1() - 6, getW(), getH());

}

/\*\*

\* Calculates the new Y moving angle of laser by converting the degrees to

\* radians and then using sine to get the vertical distance

\*

\* @return double laser Y movingAngle

\*/

public double laserYMovingAngle(double movingAngle) {

// moves a y-angle with the sin method of the laser

return (double) (Math.sin(laserAngle \* Math.PI / 180));

}

/\*\*

\* Calculates the new X moving angle of laser by converting the degrees to

\* radians and then using cosine to get the horizontal distance

\*

\* @return double laser X movingAngle

\*/

public double laserXMovingAngle(double movingAngle) {

// moves a x-angle with the cosine method of the laser

return (double) (Math.cos(laserAngle \* Math.PI / 180));

}

/\*\*

\* Updates the X and Y position of the laser. Lasers will disappear from screen

\* if they touch window border

\*/

public void move() {

if (this.laserOnScreen) {

this.newXPosition(this.getXVelocity());

if (this.getlaserX1() < 0 | this.getlaserX1() > widthOfWindow) {

this.laserOnScreen = false;

}

this.newYPosition(this.getYVelocity());

if (this.getlaserY1() < 0 | this.getlaserY1() > heightOfWindow) {

this.laserOnScreen = false;

}

}

}

}